

APPLICATION
FOR
UNITED STATES LETTERS PATENT

TITLE: PROGRAM UPDATE APPARATUS AND METHOD

APPLICANT: Kenichi MURAI and Mitsuaki HIRONO

"EXPRESS MAIL" Mailing Label Number: EV049244269US
Date of Deposit: March 5, 2002



22511

PATENT TRADEMARK OFFICE

PROGRAM UPDATE APPARATUS AND METHOD

Background of Invention

[0001] 本発明は、プログラム更新装置およびプログラム更新方法に関する。

特に、本発明は、機器組込みシステムのバグフィックスやバージョンアップを行なう際に用いられるプログラム更新装置およびその更新方法に関するものである。

[0002] 機器組込みシステム等のプログラムでは、バグを解消するためのバグフィックスや、プログラムの改良、データの更新などに伴うバージョンアップが必要となることがある。そのため、従来からも機器組込みシステム等のプログラムの書き換えが行なわれることがある。

[0003] 図1はエネルギー使用量を計測する家庭用の使用量計測器の場合を表している。使用量計測器1は家屋の外壁等に設置されており、家庭2で 사용되는機器3には、第1のエネルギー供給ライン4や第2のエネルギー供給ライン5等を通じ、使用量計測器1を経由してエネルギーが供給されている。例えば、第1のエネルギー供給ライン4を通じて供給されるエネルギーとは電気のことであり、使用量計測器1はその電気使用量を計測し、積算している。また、第2のエネルギー供給ライン5を通じて供給されるエネルギーとはガスのことであり、使用量計測器1はそのガス使用量を計測し、積算している。

[0004] このような使用量計測器の目的は、電気やガス等のエネルギーの使用量を計測することであるが、これらのデータは料金徴収の目的以外にも使用されている。例えば、エネルギー供給元にとっては、1日ないし1週間の各時間ごとの細かい消費傾向や季節ごとの消費傾向を知ることができれば、そ

のデータに基づき、エネルギー消費者に対して各時間毎の細かな料金設定を行なうことが可能になり、サービスの向上を図ることができる。また、使用ピークの時間帯には高いエネルギー使用料金を設定することで、エネルギーの総消費量を抑制し、エネルギー消費が特定の時間帯に集中してエネルギーの供給不足になるのを防止することができる。

[0005] 上記のように料金徴収やデータ活用のためには、エネルギー供給元は、各家庭の使用量計測器を検針してエネルギー使用量を調べる必要がある。そのため、管理センタ（電気会社やガス会社等のエネルギー供給元）から派遣される計測人（検針人）が各家庭を訪問し、使用量計測器のメータを検針している。しかし、このようにして人手作業によって各家庭の使用量計測器を検針するという方法では、1日に検針できる戸数が限られており、人件費もかさみ、非常に効率が悪かった。

[0006] このようなことから、家庭に設置された使用量計測器を遠隔地から自動検針できるようにしたものがある。このような従来例では、図2に示すように、電柱8の上などにデータ収集器9を設置しておき、データ収集器9を中継して各家庭2の使用量計測器1を管理センタ6につないでいる。データ収集器9とは、一定時間毎に各家庭2の使用量計測器1からデータを収集し、そのデータを管理センタ6へ送る機能を持った機器である。データ収集器9は、電源の供給に困らない、電波的に遮断物が少ない、人に触れて欲しくないといったことから、電柱8の上など人の行きにくいところに設置されている。各家庭2の使用量計測器1や管理センタ6とデータ収集器9との間は、通信回線10、11などによって結ばれる。

[0007] このような方式によれば、多くの家庭の使用量計測器を遠隔地（管理センタ6）から瞬時に自動点検することができ、人件費の削減を図れ、また

1日に検針できる件数も制限が無くなり、効率的に検針を行なうことができるようになっている。

[0008] また、別な例として、家庭内に設置されている機器を外部から操作する場合を考える。図3に示す例では、家庭2内に設置されている機器12A、12B、...と機器制御器14とを信号線13を通じて結んでおき、電話回線やネットワーク等の通信回線15に接続された機器制御器14を経由して各機器12A、12B、...を家庭2の外から操作したり、機器12A、12B、...からの情報を外部で取得できるようにしている。例えば、機器12Aが監視カメラであれば、電話回線に接続したモバイルコンピュータや携帯電話により外部から監視カメラの画像をモニターしたり、監視カメラの撮影方向を操作したりすることができ、機器12Bがビデオレコーダであれば、ビデオレコーダの録画予約を外部から行なったり、機器12Cがホームセキュリティに必要な監視センサであれば、遠隔地から外部にいてセキュリティシステムからの通知を取得することができる。

[0009] 機器制御器14に接続される機器にはさまざまなものが想定され、機器ごとに持っている情報が違うため、機器制御器14に機器を接続するにあたっては、その機器に応じて機器制御器14のシステムプログラムを書き換える必要がある。

[0010] また、ホームセキュリティに関連するシステムで用いられている場合には、ホームセキュリティ動作中では機器制御器14がフリーズすることは許されない。フリーズするとは、機器の応答が無くなり、外部へ状態を通知できなくなったり、外部からの入力を受け付けない状態になることである。ホームセキュリティとは、不審者の侵入、火災、ガス漏れ、非常通報、緊急・救急の検知をするものであり、機器制御器14がフリーズすると、ホー

ムセキュリティが意味をなさず、機器制御器 1 4 の信頼性が損なわれる問題がある。

[0011] 通信回線を通じて外部と結ばれた機器組込みシステム、例えば上記のようなデータ収集器や機器制御器における技術的課題を説明する。

[0012] データ収集器 9 や機器制御器 1 4 においては、システム（動作プログラム）にバグが見つかることがあり、そのような場合にはシステムプログラムのバグフィックスが必要になる。また、データ収集器 9 や機器制御器 1 4 に新しい機器（例えば、使用量計測器 1 など）を追加接続した場合や、データ収集器 9 や機器制御器 1 4 に新しい機能を追加したい場合には、それぞれのシステムプログラムにも対応する機能を追加したり、システムプログラムをアップデート（更新）したりする必要がある。そして、バグフィックスや機能追加のためにプログラムをアップデートするためには、データ収集器 9 や機器制御器 1 4 に組み込まれているシステムのプログラムを書き換える必要がある。

[0013] このようなプログラムの書き換えが必要となる場合、例えばアップデート用のプログラムを提供してユーザーサイドで書き換えをして貰うということになると、機械に苦手なユーザーではプログラムの書き換えやフリーズした際の復旧作業を行なって貰うことは期待できない。また、個々のデータ収集器 9 や機器制御器 1 4 はプログラム提供者（管理センタなど）から遠隔地にあることが多く、しかも、データ収集器 9 や機器制御器 1 4 では人の行きにくい場所（電柱の上など）に設置されていることが多い。よって、システムプログラムの書き換えを行なう場合には、人の手を介さず、通信回線等を通じて、自動的に書き換えを行なう必要がある。

[0014] さらに、ホームセキュリティに関連した機器制御器 1 4 等では、書き換えのためにシステムを停止させることはできず、システムを稼働させた状

況下で書き換えを行なう必要があり、書き換え中にシステムがフリーズしたような場合でも、とりあえず再起動させてシステムを稼働させる必要がある。

[0015] よって、機器組込みシステムによっては、通信回線等を通じて自動的にプログラムの書き換えを行えると共に万一の場合にもシステムがフリーズすることのないアップデート方法が求められている。

[0016] 従来でも、通信回線を通じて機器組込みシステムのプログラムを書き換える処理は行なわれている。しかし、それは、正しく動くプログラムが前提で、それが正しく書き込めたか否かをチェックするだけであった。そのため、致命的な部分は書き換えないようにしている。また、プログラムのチェックにチェックサムのみを利用している場合には、複数のエラーが発生した場合、チェックする値が偶然正しい値に一致する可能性があり、正しく書き込めたと誤認識する恐れがあった。

[0017] さらに、従来の書き換え方式では、バグのあるプログラムには対応できず、オペレーティングシステムが動作しなければ、ネットワークダウンロードなど複雑な処理は行えなかった。

Summary of Invention

[0018] 本発明の目的とするところは、機器組込みシステム（特に、常時動作する必要のある機器のシステム）において、自動的にシステムのアップデートを行え、しかもシステムをフリーズさせることのないプログラム更新装置およびその更新方法を提供することにある。

[0019] 本発明にかかるプログラム更新装置は、メモリ機器又は通信回線を通じて動作プログラムをアップデートするためのプログラム更新装置であって、動作プログラムが格納されている第1の記憶領域とは別の第2の記憶領域を確保する手段と、第1の記憶領域に格納されている第1の動作プログラムと

は別の第2の動作プログラムを第2の記憶領域に格納する手段と、第1の動作プログラム又は第2の動作プログラムのいずれかを選択的に起動させる選択起動手段と、起動を選択された動作プログラム又は当該動作プログラムの上で動作するアプリケーションプログラムの起動の失敗を検知する失敗検知手段と、前記アプリケーションプログラムの動作に必要な設定を変更する修復アプリケーションとを備え、前記失敗検知手段が起動を選択された動作プログラムの起動の失敗を検知した場合には、別の動作プログラムを選択して起動させ、前記失敗検知手段が前記アプリケーションプログラムの起動の失敗を検知した場合には、修復アプリケーションにより、前記アプリケーションプログラムの動作に必要な設定を修復させるようにしたことを特徴としている。ここで、動作プログラムとは、機器とアプリケーションを制御するプログラムであって、オペレーティングシステム（OS）も含まれる。また、メモリ機器とは、プログラム更新装置に接続することのできるメモリであれば特に限定されるものでなく、メモリカード、メモリスティックなどがある。通信回線は、有線でも無線でもよく、例えばインターネットや携帯電話、PHS（Personal Handy-phone System）によるネットワーク（通信網）を用いることができる。

[0020] このプログラム更新装置にあつては、第1及び第2の記憶領域のうち一方の記憶領域にそれまで動作していた動作プログラムを残したままで、他方の記憶領域に新しい動作プログラムをダウンロードし、第1の記憶領域に格納されている動作プログラムと第2の記憶領域に格納されている動作プログラムのうち、いずれかを選択的に起動させることができるので、いずれか一方の記憶領域に新しい動作プログラムをダウンロードした後、その動作プログラムの起動に失敗した場合には、他方の記憶領域に保存されている以前の動作プログラムを再起動させることができ、動作プログラムのアップデートに失敗しても動作プログラムを組み込まれている機器がフリーズするのを回

避できる。さらに、動作プログラムが起動したとしても、新しい動作プログラムとアプリケーションプログラムの相性が悪くアプリケーションプログラムが動作しないときであっても、アプリケーションが動作していないことを検出し、アプリケーションプログラムを動作させるための設定を修復アプリケーションが行い、アプリケーションプログラムを動作させることができる。また、このプログラム更新装置にあっては、通信回線を介して遠隔から自動的にアップデートを行なわせることができる。従って、このプログラム更新装置によれば、遠隔から自動的に動作プログラムをアップデートさせることができ、しかもアップデートに失敗した場合でもシステムがフリーズするのを防止することができる。

[0021] また、前記プログラム更新装置においては、プログラム更新装置の外部に通知する通知手段をさらに備え、前記動作プログラムのアップデートが正常になされなかった場合には、前記通知手段により、アップデートが正常になされなかった情報を前記プログラム更新装置の外部に通知させるようにすることが望ましい。ここで通知手段とは、通信回線を使って出力する手段であっても良いし、その旨を知らせる表示装置、またはその両方であってもよい。このような実施形態によれば、アップデートに失敗して動作プログラムが更新されずに動作を続けている場合に、プログラム更新装置からアップデートのデータを送る側に通知することにより、再度アップデート作業を試みる事が可能となる。

Brief Description of Drawings

[0022] 図1はエネルギー使用量を計測する家庭用の使用量計測器の使用状態を示す図である。

[0023] 図 2 は家庭に設置された使用量計測器を遠隔地から自動検針できるようにした構成を説明する図である。

[0024] 図 3 は家庭内に設置されている機器を外部から操作するための機器制御器を説明する図である。

[0025] 図 4 は本発明に係るプログラム更新方式の実行手順を示すフロー図である。

[0026] 図 5 は図 4 の復旧起動ステップの詳細を示すフロー図である。

[0027] 図 6 は図 4 のプログラム更新方式を具体的に説明する図である。

[0028] 図 7 は本発明に係るプログラム更新方式を実行するためのシステム構成を示す図である。

[0029] 図 8 はブートステータスレジスタの構造を示す図である。

[0030] 図 9 は図 7 に示したシステムを機能として表現した図である。

[0031] 図 10 は図 7 ～図 9 のシステム構成におけるアップデート方法の手順を示す図である。

[0032] 図 11 は OS 上で J a v a VM が動いている J a v a 実行環境の一例を示す図である。

[0033] 図 12 は動作プログラムをダウンロードしてシステムをアップグレードする際の全体の流れを表した図である。

[0034] 図 13 は図 12 の続図である。

[0035] 図 14 は正常解決機能における起動プログラム決定部の処理を説明する図である。

[0036] 図 15 は復旧挑戦機能における起動プログラム決定部の処理を説明する図である。

[0037] 図 1 6 は确实復旧機能における起動プログラム決定部の処理を説明する図である。

[0038] 図 1 7 は正常解決機能及び復旧挑戦機能における起動アプリケーション決定部の処理を説明する図である。

[0039] 図 1 8 は确实復旧機能における起動アプリケーション決定部の処理を説明する図である。

[0040] 図 1 9 は正常解決機能、復旧挑戦機能、确实復旧機能におけるアプリケーション動作監視部の処理を説明する図である。

[0041] 図 2 0 は正常解決機能におけるソフトウェアの流れを示すフロー図である。

[0042] 図 2 1 は復旧挑戦機能におけるソフトウェアの流れを示す図である。

[0043] 図 2 2 は第 1 の确实復旧機能におけるソフトウェアの流れを示す図である。

[0044] 図 2 3 は第 2 の确实復旧機能におけるソフトウェアの流れを示す図である。

[0045] 図 2 4 は自動検針システム（データ収集器）のモジュール構成図である。

[0046] 図 2 5 は図 2 4 の自動検針システムのハードウェア構成図である。

[0047] 図 2 6 は図 2 5 の ROM とコンパクトフラッシュメモリの状態を示す図である。

Detailed Description

[0048] 本発明に係るプログラム更新方式は、図4に示すように大まかには、ダウンロードステップ（ステップS22）、新起動ステップ（ステップS23）、復旧起動ステップ（ステップS25）からなる。そして、システムのアップデートが開始すると（ステップS21）、システムに新動作プログラム（アップデートされた動作プログラム）をダウンロードし、システム内のプログラム動作とは別領域にある記憶装置空き領域（例えば、複数のROM領域において、旧動作プログラムが稼働中でないROM領域）に新動作プログラムを転送し、書き込みを行なう（ダウンロードステップ、ステップS22）。新動作プログラムがダウンロードされて転送と書き込みが終了したら、ダウンロードの時点で動作していた旧動作プログラムに代えてダウンロードした新動作プログラムを起動させる（新起動ステップ、ステップS23）。そして、新動作プログラムが正常に起動したか否かを判定し（ステップS24）、新動作プログラムが正常に起動していたら、正常解決機能によってアップデートが完了したと判断し、アップデート作業を完了する（ステップS26）。

[0049] これに対し、ステップS24において、新動作プログラムが正常に起動していないと判断された場合には、復旧挑戦機能および确实復旧機能を内容とするシステム復旧動作を行なってシステムを再起動させてフリーズから復旧させ（復旧起動ステップ、ステップS25）、アップデート作業を完了する（ステップS26）。このとき、新動作プログラムにアップデートされている場合と、确实に動作させるために旧動作プログラムに戻っている場合がある。旧動作プログラムのまま動作する場合には、プログラム更新装置の外部に通信回線又は表示等によって通知し、次の新動作プログラムのアップデート作業を待つ。

[0050] ここで上記正常解決機能とは、ダウンロードした新動作プログラムが正常起動し、システムの動作プログラムが旧動作プログラムから新動作プログラムに切り換わってシステムが新動作プログラムで動作するようにする機能であり、復旧挑戦機能および確実復旧機能とは、ダウンロードした新動作プログラム等が正常起動できない場合に、システムがフリーズしないよう復旧させるための機能である。

[0051] しかして、本発明に係るプログラム更新方式では、旧動作プログラムが稼働しているシステムに新動作プログラムをダウンロードし、稼働するプログラムを旧動作プログラムから新動作プログラムに切り替えて新動作プログラムを実行させる。そして、フリーズ等により新動作プログラムの起動に失敗した場合には、もとの旧動作プログラムを再起動し、旧プログラムを実行させる。あるいは、旧動作プログラム又は新動作プログラムが稼働している状態で、実行アプリケーションがフリーズ等により実行停止した場合には、修復アプリケーションを実行して実行アプリケーションを修復させるものである。ここで、動作プログラムとは機器とアプリケーションを制御するプログラムであって、オペレーティングシステム（OS）も含まれる。また、旧動作プログラムとは、ダウンロード実行時に稼働中の動作プログラムであり、新動作プログラムとは、置換用の動作プログラム（アップデートプログラム）である。実行アプリケーションとは、動作プログラム上で動く、機器の利用目的を実行するアプリケーションプログラムである。修復アプリケーションとは、動作プログラム上で動く、修復機能を持った必要最低限のアプリケーションプログラムであり、動作プログラム上でアプリケーションプログラムを動作させるための設定を行う。このアップデート作業においては、遠隔からプログラムのアップデートを行なうため、通信回線を通じてプログラムをダウンロードする。通信回線は、有線でも無線でもよく、例えばインタ

一ネットや携帯電話、PHSによるネットワーク（通信網）を用いることができる。

[0052] なお、システムのフリーズは、(1)ダウンロードした動作プログラムが破壊されている場合、(2)初期化データなどのデータが破壊されている場合、あるいは、(3)動作プログラムは正常に動作していても実行アプリケーションが正常に起動しない場合に起きるが、本発明のアップデート方式では、以下に説明するようにしてシステムのフリーズを回避する。

[0053] 図5はシステム復旧動作（復旧起動ステップ、ステップS 2 5）の詳細なフローである。新動作プログラムが正常起動しない場合には、復旧起動ステップを開始する（ステップS 3 1）。まず新動作プログラムが正常動作しているかどうか判断し（ステップS 3 2）、新動作プログラムが正常動作している場合には、実行アプリケーションが動作していないので、第1の确实復旧機能で修復アプリケーションを起動させ、実行アプリケーションを実行可能な状態に修復し（ステップS 3 3）、終了する（ステップS 3 8）。ステップS 3 2で新動作プログラムが正常動作していなければ、復旧挑戦機能で旧動作プログラムからの立ち上げを行う（ステップS 3 4）。旧動作プログラムは、このアップデート動作まで動いていたプログラムであるから、間違いなく起動する。その旧動作プログラムの上で実行アプリケーションが起動するか判断し（ステップS 3 5）、実行アプリケーションが起動すれば、旧動作プログラムのままで動作している旨の通知を出し（ステップS 3 7）、終了する（ステップS 3 8）。旧動作プログラムの上で実行アプリケーションが起動しない場合には、新動作プログラムによって起動のための設定が変更されている可能性があり、第2の确实復旧機能で修復アプリケーションを起動させ、実行アプリケーションを実行可能な状態に設定を修復し（ステッ

プS 3 6)、旧動作プログラムのままで動作している旨の通知を出し(ステップS 3 7)、終了する(ステップS 3 8)。

[0054] 図6は上記方式を具体的に説明する図である。図6(a)はダウンロード前の状態を表しており、システム内では旧動作プログラム21により実行アプリケーション22が動作している。なお、図6において、符号「RUN」は動作中のアプリケーションを表している。ついで、アップデート作業が開始すると、図6(b)に示すように、ネットワーク等の通信回線又はメモリーカード等を通じてシステムにデータ25がダウンロードされ、図6(c)に示すように新動作プログラム24が空き領域(例えば、複数のROM領域ROM1とROM2があり、ROM1の領域が稼働中であるとすれば、ROM2の領域)に転送され、そこに書き込まれる。新動作プログラム24の書き込みが終了すると、図6(d)に示すように、正常解決機能によって書き込まれた領域(例えば、上記ROM2の領域)から新動作プログラム24が起動され、動作プログラムが旧動作プログラム21から新動作プログラム24に切り替えられ、実行アプリケーション22が新動作プログラム24の上で実行され、修復アプリケーション23も新動作プログラム24の上で実行可能となる。

[0055] 新動作プログラム24を起動すると(新起動ステップ)、新動作プログラム24の起動に成功(OK)したか、失敗(NG)したか判断し、新動作プログラム24のダウンロードが正常完了していて新動作プログラム24の起動に成功していれば(図6(e))、アップデート機能が完了する(図6(f))。

[0056] このようにして正常解決機能が働いてダウンロードした新動作プログラム24でシステムが正常に起動されると、機器の目的を妨げることなく(すなわち、システムを停止させることなく)システムの書き換えが完了し、

機器の目的を実行しつつ、同時にシステムの書き換えを行なうという本発明の目的が達成される。

[0057] これに対し、新動作プログラム 2 4 の起動に失敗した場合には、失敗の状況に応じて復旧挑戦機能、第 1 又は第 2 の確実復旧機能の 3 種類のシステム復旧機能のうちから最適な方法を実行する。すなわち、復旧挑戦機能とは、正常解決機能による新動作プログラム 2 4 の起動に失敗した場合に、図 6 (g) に示すように、ダウンロードした新動作プログラム 2 4 を実行せず、元の旧動作プログラム 2 1 を再起動して旧動作プログラム 2 1 でシステムを実行し直すものである。また、第 1 の確実復旧機能とは、正常解決機能により新動作プログラム 2 4 が正常に起動されても、実行アプリケーション 2 2 の起動に失敗した場合、図 6 (h) に示すように、ダウンロードした新動作プログラム 2 4 が起動された状況において、修復アプリケーション 2 3 を実行して実行アプリケーション 2 2 を修復するものである。第 2 の確実復旧機能とは、正常解決機能による新動作プログラム 2 4 の起動に失敗した後、復旧挑戦機能によって旧動作プログラム 2 1 が復旧されたが、実行アプリケーション 2 2 の起動に失敗した場合に、図 6 (i) に示すように、旧動作プログラム 2 1 が実行されている状況において、修復アプリケーション 2 3 を実行して実行アプリケーション 2 2 を修復するものである。ここで修復アプリケーションは、旧動作プログラム 2 1 又は新動作プログラム 2 4 の上で実行アプリケーション 2 2 が動作できるように設定を調整する。

[0058] 新動作プログラム 2 4 の起動失敗の状況が、新動作プログラム 2 4 は正常に起動したが、実行アプリケーション 2 2 がフリーズした場合には、第 1 の確実復旧機能を働かせて実行アプリケーション 2 2 を修復し（図 6 (h) ）、新動作プログラム 2 4 の上で実行アプリケーション 2 2 を動作させ、アップデート機能が完了する（図 6 (f) ）。また、新動作プログラム

24がフリーズした場合には、復旧挑戦機能を働かせて動作プログラムを新動作プログラム24から旧動作プログラム21に戻して旧動作プログラム21を再起動し（図6（g））、システムが正常に動くようになれば修復を完了し（図6（j））、旧動作プログラム21の上で実行アプリケーション22を動作させ、アップデート機能が完了する（図6（f））。もし、復旧挑戦機能を作動させてもシステムが復旧されなかった場合には、すなわち旧動作プログラム21は正常に再起動したが、実行アプリケーション22がフリーズした場合には、第2の確実復旧機能を働かせて旧動作プログラム21の上で修復アプリケーション23を実行して実行アプリケーション22を修復し（図6（i））、旧動作プログラム21の上で実行アプリケーション22を動作させ、アップデート機能が完了する（図6（f））。

[0059] このようにして正常解決機能が失敗した場合でも、復旧挑戦機能によって旧動作プログラムでシステムを再起動させることができ、あるいは確実復旧機能によって修復アプリケーションで実行アプリケーションを修復することができるので、決してシステムがフリーズすることが無く、新動作プログラムの起動に失敗してもシステムをフリーズさせないという本発明の目的が達成される。なお、アップデート機能がフリーズすることなく完了するが、ここで完全に新動作プログラムの置き換えの目的を達成しているのは図6（d）（h）である。図6（g）（i）は旧実行プログラムで動作し続けており、この場合プログラム更新装置は新実行プログラムのダウンロード元に、新動作プログラムが動作していない旨の通知を出し、再度ダウンロードを要求しても良いし、次の新実行プログラムのダウンロードを待っても良い。

[0060] 次に、上記のようなリモートプログラム更新方式を実行するためのシステム構成を図7に示す。このシステムはデータ収集器や機器制御器等の機器に組み込まれており、マイクロプロセッサ（CPU）26、パワーオンリ

セット (Power On Reset) 回路 27、ウォッチドックタイマ (WDT) 28、ブートステータスレジスタ 29、ブートプログラム ROM 30、プログラム ROM 31、32、RAM 33、入出力部 (I/O) 34、ネットワーク 35、メモリカード 36 等によって構成され、これらはバスライン 37 を通じて結ばれている。ここで、マイクロプロセッサ 26 は、システム全体を制御するものである。パワーオンリセット回路 27 は、システムをオン、オフし、あるいはハードウェアリセットを掛けるための回路である。ウォッチドックタイマ 28 は、システムがタイムアウトした際にソフトウェアリセットを掛けることができるものである。ブートプログラム ROM 30 は、最小限のブートプログラム領域である。プログラム ROM 31、32 は、2 重化されたプログラム格納領域である。ブートプログラム ROM 30 及びプログラム ROM 31、32 は、同一メモリの領域を分割したものであってもよい。RAM 33 は、正常解決機能等を実行する際など、一時的にデータを保存するものである。ネットワーク 35 は、インターネット等を通じて外部 (公衆回線等) と結ばれており、新動作プログラムはネットワーク 35 を通じていずれか一方のプログラム ROM 31、32 にダウンロードされる。また、新動作プログラム 24 は、入出力部 34 やメモリカード 36 からインストールできるようにしてもよい。

[0061] ブートステータスレジスタ 29 は、起動させた動作プログラムや実行アプリケーションが異常となったかどうかを記憶しておくレジスタであって、ソフトウェアリセットによっては値が変更されない。ブートステータスレジスタ 29 は、図 8 に示すように、最下位ビットに BOOT フラグを持ち、その上位ビットに CONFIG フラグを持っている。BOOT フラグは、電源オンの直後には "0" となっているが、初回ブートイメージ異常があると値 "1" が保存される。また、CONFIG フラグは、電源オンの直後には "0" となっているが、初期設定値に異常があると値 "1" が保存される。システムは、

新動作プログラム 24 を起動した際、新動作プログラム 24 の異常でソフトウェアリセットがかかると、ブートステータスレジスタ 29 の BOOT フラグをチェックし、旧動作プログラム 21 が格納されている ROM 領域と新動作プログラム 24 が格納されている ROM 領域のうちいずれの ROM 領域から動作プログラムを起動するかを決定する。また、実行アプリケーション 22 の異常でソフトウェアリセットがかかると、システムは、ブートステータスレジスタ 29 の CONF 1 G フラグをチェックし、実行アプリケーション 22 と修復アプリケーション 23 のうちどちらのアプリケーションを起動するか決定する。

[0062] 図 9 は図 7 に示したシステムが各プログラムによって動作する機能を表現したものである。これらの機能は、

(f 1) ダウンロードされた新動作プログラム 24 の格納場所（メモリ領域）を決定する機能、

(f 2) 決定された格納場所へダウンロードされた新動作プログラム 24 を格納する機能、

(f 3) 格納された新動作プログラム 24 をチェックする機能、

(f 4) 格納された新動作プログラム 24 を起動する機能、

(f 5) ダウンロードする前に動作していた旧動作プログラム 21 を起動する機能、

(f 6) 起動された新動作プログラム 24 または旧動作プログラム 21 の動作をチェックする機能、

(f 7) チェックされた動作プログラム上で実行アプリケーション 22 を起動する機能、

(f 8) チェックされた動作プログラム上で修復アプリケーション 23 を起動する機能、

(f 9) 起動された実行アプリケーション 22 または修復アプリケーション

ン 2 3 の動作をチェックする機能、
から構成されており、新動作プログラム 2 4 のダウンロードから始まって図
9 の矢印の順に実行され、アップデート機能の完了によって終了する。

[0063] しかして、図 7 ～図 9 のようなシステム構成において、本発明のリモートプログラム更新方式は図 1 0 に示す手順に沿って実行される。以下、図 9 の機能を参照しながら、図 5 に示したフローを詳細に、図 1 0 に沿ってアップデートの手順を説明する。なお、図 5 において示した、旧動作プログラムのままである旨の通知（ステップ S 3 7）は付加機能であるため、図 1 0 では省略してある。新動作プログラム 2 4 がネットワーク 3 5 を通じて外部からダウンロードされ、いずれかの ROM に格納されると（ステップ S 4 1、機能 f 1、f 2）、システムは新動作プログラムが正常に起動したかどうかチェックする（ステップ S 4 2）。すなわち、新動作プログラム 2 4 がダウンロードされると、システムは、格納された新動作プログラム 2 4 をチェックし（機能 f 3）、異常がなければ新動作プログラム 2 4 を起動し（機能 f 4）、さらに、ブートステータスレジスタ 2 9 をチェックすることにより、起動された新動作プログラム 2 4 が正常に起動したかどうかチェックする（機能 f 6）。

[0064] 新動作プログラム 2 4 が正常に起動している場合には、起動された新動作プログラム 2 4 上で実行アプリケーション 2 2 を起動させ（機能 f 7）、実行アプリケーションが正常に起動したかどうかチェックする（ステップ S 4 3）。すなわち、システムは、ブートステータスレジスタ 2 9 をチェックすることによって実行アプリケーションに異常がないかチェックする（機能 f 9）。そして、実行アプリケーションが正常に起動していれば、システムの起動を完了する（アップデート完了、ステップ S 4 5）。

[0065] これに対し、ステップS 4 3で実行アプリケーション2 2に異常があった場合には、ブートステータスレジスタ2 9のCONF I Gフラグが”1”になっているので、ウォッチドックタイマ2 8のオーバertimeによってソフトウェアリセットが掛かり、ブートステータスレジスタ2 9のCONF I Gフラグがチェックされると、修復アプリケーション2 3が起動される。起動された修復アプリケーション2 3はシステム（実行アプリケーション2 2）の設定を修復し（ステップS 4 4、機能f 8）、システムの起動を完了する（ステップS 4 5）。

[0066] また、ステップS 4 2で新動作プログラム2 4が異常である場合には、ブートステータスレジスタ2 9のBOOTフラグが”1”になっているので、ウォッチドックタイマ2 8のオーバertimeによってソフトウェアリセットが掛かり、ブートステータスレジスタ2 9のBOOTフラグがチェックされると、旧動作プログラム2 1でシステムを再起動してシステムを旧動作プログラム2 1に復旧させる（ステップS 4 6、機能f 5）。ついで、旧動作プログラム2 1上で実行アプリケーション2 2を起動させ（機能f 7）、実行アプリケーション2 2が正常に起動したかどうかチェックする（ステップS 4 7）。そして、実行アプリケーション2 2が正常に起動していれば、システムの起動を完了する（アップデート完了、ステップS 4 5）。これに対し、実行アプリケーション2 2が異常である場合には、ウォッチドックタイマ2 8のオーバertimeによってソフトウェアリセットが掛かり、ブートステータスレジスタ2 9のCONF I Gフラグがチェックされると、修復アプリケーション2 3を起動してシステム（実行アプリケーション2 2）の設定を修復し（ステップS 4 8、機能f 8）、システムの起動を完了する（ステップS 4 5）。

[0067] 次に、OS（オペレーティングシステム）上でJava VM（仮想マシン）が動いているJava実行環境下においてアップデートが行なわれる様子を具体的に説明する。図11はOS上でJava VMが動いているJava実行環境の一例を示しており、システム起動時に働くブートプログラム41、システム全体を制御するOS42、OS42の違いを吸収して共通のプラットフォームを提供するJava VM43、Java VM上で動くクラスライブラリ44、機器の利用目的を実行する実行アプリケーション46、必要最小限の修復機能を持った修復アプリケーション47によって構成されている。ここで、OS42、Java VM43及びクラスライブラリ44がアップデート対象となる動作プログラム45である。以下説明するために、アップデート対象部分を動作プログラムに限定しているが、2重化したメモリで動作プログラムを切り替えるので、同じメモリ上にある実行アプリケーションもアップデートの対象になることは言うまでもない。

[0068] 一方、上記動作プログラム45をアップデートするためのアップデート実行部は、新しいJava VM及びOSと以前からのJava VM及びOSのうちいずれの動作プログラムを起動させるかを決める起動プログラム決定部48、どのアプリケーションプログラムを起動させるかを決定する起動アプリケーション決定部49、起動したアプリケーションの動作を監視するアプリケーション動作監視部50を備えている。

[0069] この環境下におけるアップデート方式でも、正常解決機能、復旧挑戦機能、第1及び第2の確実復旧機能が実現されているが、まず、これらの機能の詳細は省き、動作プログラムをダウンロードしてシステムをアップグレードする際の全体の流れを図12及び図13により説明する。このシステムでは、新しいJava VM及びOS（新動作プログラム）をネットワーク35等を通じて外部からダウンロードすると（ステップS51）、ダウンロー

ドされた J a v a VM及びOSは、現在動作している J a v a VM及びOS（旧動作プログラム）により使用されていないROM領域に格納される（ステップS 5 2）。ついで、システムにハードウェアリセットまたはソフトウェアリセットを掛け、システムを再起動する（ステップS 5 3）。その際、システムの起動プログラム決定部4 8は、ブートステータスレジスタ2 9をチェックすることにより、新旧いずれの J a v a VM及びOSを起動させるかを決定する（ステップS 5 4）。起動する J a v a VM及びOSが決まると、システムは決定された J a v a VM及びOSが正常か否かをチェックサムを用いてチェックする（ステップS 5 5）。このチェックにより、決定された J a v a VM及びOSが正常でないと判断されると、起動する J a v a VM及びOSを他方の J a v a VM及びOSに変更する（ステップS 5 6）。

[0070] こうして正常に起動できる J a v a VM及びOSが決定されると、ウォッチドックタイマ2 8が起動し（ステップS 5 7）、ブートステータスレジスタ2 9のBOOTフラグを反転させ（ステップS 5 8）、J a v a VM及びOSを起動する（ステップS 5 9）。

[0071] ついで、起動アプリケーション決定部4 9が、起動させるアプリケーションを決定すると（ステップS 6 0）、決定したアプリケーションをロードし（ステップS 6 1）、ブートステータスレジスタ2 9のBOOTフラグとCONF I Gフラグを反転させる（S 6 2）。これによって、ステップS 5 8で反転されたBOOTフラグが元に戻る。また、この段階に達するまでに J a v a VM及びOSがフリーズしていれば、ウォッチドックタイマ2 8のタイムアウトでソフトウェアリセットがかかっている筈であるから、この時点で、J a v a VM及びOSが正常に起動したと判断できる。従って、ウォッチドックタイマ2 8を更新し（ステップS 6 3）、アプリケーション動

作監視部50により、ロードしたアプリケーションの動作を監視する（ステップS64）。

[0072] 図14～図19は、図12及び図13に示した処理中に行なわれる正常解決機能、復旧挑戦機能、確実復旧機能の処理手順を示している。なお、図14～図19において、共通部とは、正常解決機能、復旧挑戦機能、確実復旧機能に共通な図12、18の処理を指している。まず、図14により正常解決機能における起動プログラム決定部48の処理を説明する。図12のステップS58でブートステータスレジスタ29のBOOTフラグが反転された後、起動プログラム決定部48はブートステータスレジスタ29のBOOTフラグをチェックする（ステップS71）。そして、起動プログラム決定部48は、BOOTフラグが反転しているか否か調べ（ステップS72）、反転していれば、ダウンロードする前のJava VM+OSで起動するように決定し（ステップS74）、BOOTフラグが反転していなければダウンロードしたJava VM+OSで起動するように決定する（ステップS73）。ダウンロードしたJava VM及びOSに異常がなくて正常解決機能が働く場合には、BOOTフラグは反転していないので、起動プログラム決定部48はダウンロードしたJava VM及びOSを起動するように決定する（ステップS73）。

[0073] また、図15は復旧挑戦機能における起動プログラム決定部48の処理を説明している。図12のステップS58でブートステータスレジスタ29のBOOTフラグが反転された後、起動プログラム決定部48はブートステータスレジスタ29のBOOTフラグをチェックする（ステップS71）。そして、起動プログラム決定部48は、BOOTフラグが反転しているか否か調べ（ステップS72）、BOOTフラグが反転していれば、ダウンロードする前のJava VM+OSで起動するように決定し（ステップS74）、

BOOTフラグが反転していなければダウンロードしたJava VM+OSで起動するように決定する（ステップS73）。ダウンロードしたJava VM及びOSに異常があつて復旧挑戦機能が働く場合には、正常解決機能で動作中に、図13のステップS62に達する前にウォッチドックタイマ28がタイムアウトしてソフトウェアリセットがかかり、BOOTフラグは反転したままになるので、起動プログラム決定部48はダウンロードする前のJava VM及びOSを起動するように決定する（ステップS74）。

[0074] また、図16は确实復旧機能における起動プログラム決定部48の処理を説明している。図12のステップS58でブートステータスレジスタ29のBOOTフラグが反転された後、起動プログラム決定部48はブートステータスレジスタ29のBOOTフラグをチェックする（ステップS75）。そして、起動プログラム決定部48は、BOOTフラグが反転しているか否か調べ（ステップS76）、BOOTフラグが反転していれば、直前に動作していないJava VM+OSを起動するように決定し（ステップS78）、BOOTフラグが反転していなければ直前に動作していたJava VM+OSで起動するように決定する（ステップS77）。确实復旧機能が働く場合には、第1の确实復旧機能でも第2の确实復旧機能でもJava VM及びOSの正常起動が確認されているので、ブートステータスレジスタ29のBOOTフラグは反転しておらず、起動プログラム決定部48は直前に動作していたJava VM及びOS（第1の确实復旧機能の場合には、ダウンロードされたJava VM及びOS、第2の确实復旧機能の場合には、ダウンロードする前のJava VM及びOS）を起動するように決定する（ステップS77）。

[0075] 次に、図17により正常解決機能及び復旧挑戦機能における起動アプリケーション決定部49の処理を説明する。図13のステップS62でブー

トステータスレジスタ 29 の CONFIG フラグが反転された後、起動アプリケーション決定部 49 はブートステータスレジスタ 29 の CONFIG フラグをチェックする（ステップ S 79）。そして、起動アプリケーション決定部 49 は、CONFIG フラグが反転しているか否か調べ（ステップ S 80）、反転していれば、修復アプリケーションを起動するように CONFIG ファイル（起動時に動作するプログラムを指定する定義が書かれたファイル）を書き換え（ステップ S 81）、システムの CONFIG を初期化してプログラムを実行し（ステップ S 82）、ブートステータスレジスタ 29 の CONFIG フラグを反転させた（ステップ S 83）後、システムをソフトウェアリセットさせる（ステップ S 84）。ステップ S 80 で CONFIG フラグが反転していなければ、起動アプリケーション決定部 49 は、CONFIG フラグで設定されているアプリケーションを起動するように決定する（ステップ S 85）。正常解決機能、復旧挑戦機能が動作している場合には、CONFIG フラグは反転していないので、起動アプリケーション決定部 49 は、CONFIG フラグで設定されているアプリケーションを起動するように決定する（ステップ S 85）。

[0076] また、図 18 は确实復旧機能における起動アプリケーション決定部 49 の処理を説明している。図 13 のステップ S 62 でブートステータスレジスタ 29 の CONFIG フラグが反転された後、起動アプリケーション決定部 49 はブートステータスレジスタ 29 の CONFIG フラグをチェックする（ステップ S 79）。そして、起動アプリケーション決定部 49 は、CONFIG フラグが反転しているか否か調べ（ステップ S 80）、反転していれば、修復アプリケーションを起動するように CONFIG ファイルを書き換え（ステップ S 81）、CONFIG 初期化プログラムを実行し（ステップ S 82）、ブートステータスレジスタ 29 の CONFIG フラグを反転させた（ステップ S 83）後、システムをソフトウェアリセットさせる（ステ

ップS 8 4)。ステップS 8 0でCONF I Gフラグが反転していなければ、起動アプリケーション決定部4 9は、CONF I Gフラグで設定されているアプリケーションを起動するように決定する(ステップS 8 5)。第1又は第2の确实復旧機能が動作している場合には、正常解決機能、復旧挑戦機能でJ a v a VM+O Sの正常動作が確認されているが、アプリケーション動作監視部5 0でウォッチドックタイマ2 8がタイムアウトしたため、CONF I Gフラグが反転しているので、起動アプリケーション決定部4 9は、修復アプリケーション2 3を起動するようにCONF I Gファイルを書き換え(ステップS 8 1)、CONF I Gフラグ初期化プログラムを実行し(ステップS 8 2)、ブートステータスレジスタ2 9のCONF I Gフラグを反転させる(ステップS 8 3)。これにより、図1 2のステップS 5 8で反転されたCONF I Gフラグはもとに戻る。さらに、起動アプリケーション決定部4 9は、システムにソフトウェアリセットを掛ける(ステップS 8 4)。ソフトウェアリセットを掛けて、再度このステップへ来た時には、CONF I Gフラグはもとに戻っているので、CONF I Gファイルで設定されているアプリケーション(修復アプリケーション2 3)を起動するように決定される。

[0077] 次に、図1 9により正常解決機能、復旧挑戦機能、确实復旧機能におけるアプリケーション動作監視部5 0の処理を説明する。図1 3のステップS 6 3でウォッチドックタイマ2 8がリセットされた(ステップS 8 6)後、アプリケーション動作監視部5 0は通常処理を行ない(ステップS 8 7)、修復アプリケーション2 3が起動しているか否かチェックする(ステップS 8 8)。修復アプリケーション2 3が起動していない場合には、ウォッチドックタイマ2 8がタイムアウトしているかチェックするが、修復アプリケーション2 3が起動している場合には、ウォッチドックタイマ2 8をチェックせず、ステップS 8 6の処理へ戻ってウォッチドックタイマ2 8をリセット

する。従って、确实復旧機能が実行されている場合には、修復アプリケーション23が起動しているので、決してウォッチドックタイマ28がタイムアウトしない。これに対し、実行アプリケーション22が実行されている正常解決機能または復旧挑戦機能の場合には、アプリケーション動作監視部50はウォッチドックタイマ28のタイムアウトをチェックし（ステップS89）、ウォッチドックタイマ28がタイムアウトならソフトウェアリセットを掛けて（ステップS90）确实復旧機能へ移行し、タイムアウトしていなければ、ステップS86の処理へ戻ってウォッチドックタイマ28をリセットする。

[0078] しかして、ダウンロードしたJava VM+OS（新動作プログラム）が正常に起動し、実行アプリケーション22も正常に起動した場合（正常解決機能の場合）の処理は、図14、図17、図19の流れになる。また、ダウンロードしたJava VM+OS（新動作プログラム）が正常に起動せず、ダウンロードする前のJava VM+OS（旧動作プログラム）で正常に復旧した場合（復旧挑戦機能の場合）の処理は、図15、図17、図19の流れになる。また、正常解決機能において、ダウンロードしたJava VM+OS（新動作プログラム）は正常に起動したが、実行アプリケーション22が正常に動作せず、修復アプリケーション23で確実に復旧した場合（第1の确实復旧機能の場合）の処理は、図16、図18、図19の流れになる。また、确实復旧機能において、ダウンロードする前のJava VM+OS（旧動作プログラム）で実行アプリケーション22が正常に動作せず、修復アプリケーション23で確実に復旧した場合（第2の确实復旧機能の場合）の処理は、図16、図18、図19の流れになる。そして、正常解決機能、復旧挑戦機能、确实復旧機能の3段階を段階的に行なうことで、本発明の目的が実現される。

[0079] 次に、正常解決機能におけるソフトウェアの流れを図20に示す。この図に示すように、正常解決機能の場合には、ダウンロード部51によって新しいJava及びOS（新動作プログラム24）がダウンロードされ、現在動作していないJava VM及びOS（旧動作プログラム21）を新しいJava VM及びOSに置き換える。ついで、ブートプログラムを起動する。Java VM+OS起動／動作監視部52は、起動するJava VM及びOSを決定し、壊れていないか、フリーズしないかをチェックし、フリーズすれば、復旧挑戦機能を実行する。また、アプリケーション起動／動作監視部53は、実行アプリケーション22を起動し、フリーズしないかチェックし、フリーズしたらソフトウェアリセットを掛け、第1の確実復旧機能を実行する。

[0080] また、復旧挑戦機能におけるソフトウェアの流れを図21に示す。新しいJava VM及びOSの起動時にフリーズし、ソフトウェアリセットが掛かると、Java VM+OS起動／動作監視部52は、ダウンロードする前に動作していた古いJava VM及びOS（旧動作プログラム）を起動し直す。ついで、アプリケーション起動／動作監視部53は、実行アプリケーション22を起動し、フリーズしないかチェックする。そして、フリーズしたら、ソフトウェアリセットを掛け、第2の確実復旧機能を実行する。

[0081] また、第1の確実復旧機能におけるソフトウェアの流れを図22に示す。アプリケーション起動／動作監視部53は実行アプリケーション22を監視しており、新しいJava VM及びOSで実行アプリケーション動作時にフリーズし、ソフトウェアリセットがかかると、新しいJava VM及びOSは正常に起動することを確認しているので、Java VM+OS起動／動作監視部52により新しいJava VM及びOSを起動し直す。ついで、

アプリケーション起動／動作監視部 53 が修復アプリケーションを起動し、復旧が完了する。

[0082] また、第 2 の確実復旧機能におけるソフトウェアの流れを図 23 に示す。アプリケーション起動／動作監視部 53 は実行アプリケーション 22 を監視しており、古い J a v a VM 及び O S で実行アプリケーションが動作時にフリーズし、ソフトウェアリセットがかかると、古い J a v a VM 及び O S は正常に起動することを確認されているので、J a v a VM + O S 起動／動作監視部 52 により古い J a v a VM 及び O S を起動し直す。ついで、アプリケーション起動／動作監視部 53 は、修復アプリケーション 23 を起動し、復旧を完了する。

[0083] 図 24 は自動検針システム（データ収集器）61 のモジュール構成図である。この自動検針システム 61 は、P H S 62 を介して家庭や管理センタと定期的にデータ通信を行なっている。そして、図 24 に示すアップデート可能部分 63 が管理センタからのデータによってアップデートされるようになっている。図 25 は、図 24 の自動検針システム 61 のハードウェア構成図である。P H S 62 は P P P ドライバやシリアルドライバを介してシリアルポート 64 と接続されている。ROM 65 には、起動プログラムやブートプログラム、C O N F I G ファイルが格納される。また、外部記憶装置であるコンパクトフラッシュメモリ（C F）66 には、自動検針用アプリケーション 70、修復アプリケーション 71、自動検針により得られたデータなどが格納される。図 26 は、図 25 の ROM 65 とコンパクトフラッシュメモリ 66 の状態を示している。図 26 に示す ROM 65 の R O M 領域 67 で、起動プログラム M が立ち上がり、自動検針用アプリケーションが動作している。図 24 のアップデート可能部分 63 において、バグフィックスを行ったり、ソフト的な機能追加やハード（機器）の追加によるドライバの追加を

行なった場合には、起動プログラムMの対応する個所を修正し、コンパイルする。修正、コンパイル済みの起動プログラムをM'とすると、起動プログラムM'をネットワークを通じて、図26のROM65のROM領域68へ格納し、ROM領域68から起動し直す。また、起動プログラムM'の起動に失敗した場合には、ROM領域67のMで起動し直す。また、CONFIGファイル69のデータの破壊により自動検針用アプリケーション70の起動に失敗した場合には、図26のコンパクトフラッシュメモリ66に格納している修復アプリケーション71でデータの修復を行ない、必要最低限の機能を持ったアプリケーションで起動し直す。これにより、自動でアップデートを行え、かつ決してフリーズしないアップデートが完了する。なお、修復アプリケーションは上記に説明した、起動の設定を変更する機能のほか、telnet、FTPなどの最低限の通信機能を確保するアプリケーションであつても良いし、複数の修復アプリケーションがあつても良い。状況に応じてBOOTフラグで必要な修復アプリケーションを立ち上げるようにするとよい。たとえば、修復アプリケーションに通信機能を持つことにより、ほとんど全ての機能がフリーズした後においても、通信機能をつかって遠隔操作でフリーズを解消し、アップデートでのフリーズを防止できる。